Design and Simulation of AMBA Advanced System Bus

Ankit Berde[1]
[1]Mumbai University, India
(*E-mail:* ankitjb3@gmail.com)

***Abstract*—**AMBA specification defines protocol for Bus Architecture for data and control signals intercommunication among the processors, memory and controller interfaces. Within a high-performance embedded system, multiple processors are fused to meet the specification of the design. Thus, a fitting Bus Architecture is to be implemented for efficient inter-communication of processors. Advanced System Bus is one such flavour of AMBA transfer protocols, designed for provision of communication of a modular system. This paper outlines the implementation of Advanced System Bus Architecture for a multi-master, multi-slave communication. The design fits in a protocol involving a set of masters to write or read data from a particular slave, wherein at a time only a single master is granted the Bus to perform the transaction.

***Keywords*—**Advanced System Bus, AMBA, Verilog.

I. Nomenclature

AMBA - Advanced Microcontroller Bus Architecture

ASB - Advanced System Bus

II. Introduction

Advanced System Bus (ASB) under Advanced Microcontroller Bus Architecture specification defines a protocol for data transactions between processors, on-chip memories and off-chip memory and peripheral interfaces. ASB was first introduced by ARM Limited, and this paper presents a review on the implementation of the protocol. This paper can be used to understand the implementation challenges involved in the Simulation of the protocol. The protocol is a multi-master to multi-slave communication protocol, however, the fact remains that at a given time at the most one master can have hold over the Bus, also, at any given time it can transact with only one slave.

The paper is presented in following order:

- Section III explains the protocol for functioning of the Bus. Also, it enlightens the interface of the peripherals to the Bus.
- Section IV elaborates the design architecture and the implementation challenges.
- Section V explains the Simulation results achieved with the implementation.
- Section VI is the inferences of the simulated design.

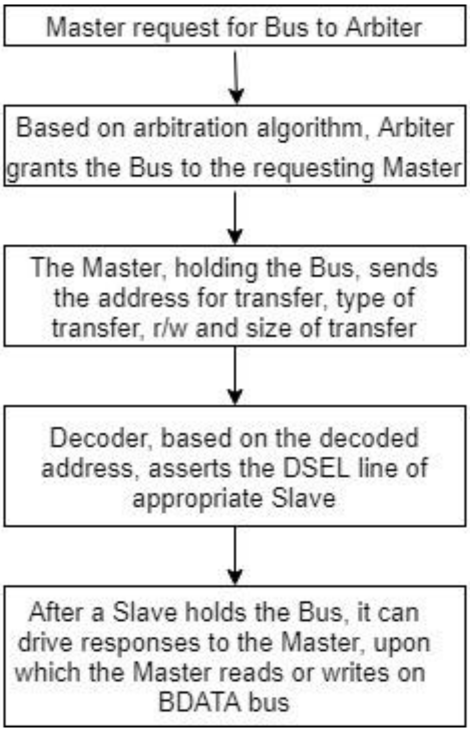
Master request for Bus to Arbiter
Based on arbitration algorithm, Arbiter grants the Bus to the requesting Master
The Master, holding the Bus, sends the address for transfer, type of transfer, r/w and size of transfer
Decoder, based on the decoded address, asserts the DSEL line of appropriate Slave
After a Slave holds the Bus, it can drive responses to the Master, upon which the Master reads or writes on BDATA bus

Fig. 1. ASB Transactions flow

III. ASB protocol

Fig 1 depicts the flow for typical transactions on ASB. After owning the Bus, the master must drive the control signals on Bus, which include type of transfer, address of transfer, r/w and size for transfer on BTRAN, BADDR, BWRITE, BSIZE pins respectively. Upon which slave responds using BWAIT, BERROR, BLAST signals after which the data is sent or received on BDATA bus by the owning master.

In case of a Burst Operation, after first Bus Transfer, the Master successively drives control and data signals, and the slaves responds to those control signals.

There are 3 distinct types of transfers on ASB:

A. *Address-only Transfer*

The Master does not access any Slave in this transaction, thus the Master is only obliged to drive a valid value for type of transfer on BTRAN pin, though the Master still has to undergo arbitration process with the Arbiter. It is a single cycle transfer and the Decoder is supposed to drive the response in association to this transfers. The purpose of these transfers is to:

- broadcast an address without committing to the transfer
- provide an IDLE cycle
- provide a turnaround cycle during bus master handover.

B. Non-sequential Transfer

Once a Master is granted the Bus through arbitration, it has to drive the control signals on the Bus. The Decoder then inserts a DECODE cycle and in case of a valid address asserts a select signal to an appropriate Slave on DSEL pin. In case of an invalid address, the Decoder responds with asserting on BERROR pin.Once a Slave owns the Bus, it drives appropriate responses to the Master. Upon that, based upon the write or read transfer, the Master or Slave respectively has to drive the data bus.

C. Sequential Transfer

The first transfer of Sequential type of transactions are always to be followed successively after the Address-only or Non-sequential Transfer. After the first transfer of the burst, the Master successively drives the control signals based upon the successive responses of the selected Slave. Upon which the master send or receives data on BDATA pin, based on the direction of transfer. In Sequential Transfer, the address of current transfer is related to that of previous transfer. If the Sequential Transfer follows a Non-sequential or another Sequential Transfer, the address of current transfer should be equal to sum of address of previous transfer and size of previous transfer. If the Sequential Transfer follows an Address-Only Transfer, then the address of the current transfer should be same as that of Address-Only cycle.

IV. IMPLEMENTATION

There are 4 major components in the ASB Architecture:

A. Master

The Master initiates any transaction on the Bus by asserting the bus request signal on its AREQ line. On grant of the Bus, the master has to drive the appropriate control signals for the transaction. The first response of the transaction is driven by the Decoder. If the driven address was legit, the Master then decodes the responses from the Slave, and as a result sends or receives the data to complete the transaction with the Slave. The Master can further keep up the transaction with successive burst transfers or release the Bus.

B. Arbiter

Based on the requests from Masters, the Arbiter asserts or de-asserts bus grant signal to the Masters. It makes sure that at any given time, a maximum of one Master owns the Bus. In our design, the Arbiter follows Round-robin arbitration algorithm to resolve the arbitration process. An assertion on AGNT line by the Arbiter of a particular Master indicates that the Master is currently the highest priority and at the completion of the current transfer, as indicated by BWAIT low, Master will be granted the bus. The Arbiter, on every cycle tracks which Master has been currently granted the Bus, so as to maintain the current Master granted in case if the respective BLOK pin is asserted and also to maintain the priority status for next bus request arbitration. The Arbiter cannot de-assert a grant signal in between of a locked transfer, as indicated by a logic high on LOK line driven by the active Master.

C. Decoder

The Decoder, based upon address of Non-sequential Transfer or first address of Sequential Transfer, drives the Decode response by indicating a logic high on BWAIT pin, and also asserts the select signal for a particular slave on DSEL pin for a valid address. For an invalid or restricted address request, Decoder issues an error response on BERROR line. In absence of any Slave owning the Bus, the Decoder abides to respond for every transfer requests by the Master. The Decoder is also responsible to drive the response for Address-Only type of transfers. Only when the Decoder is in the SLAVESEL state, the selected slave is obliged to drive the responses.

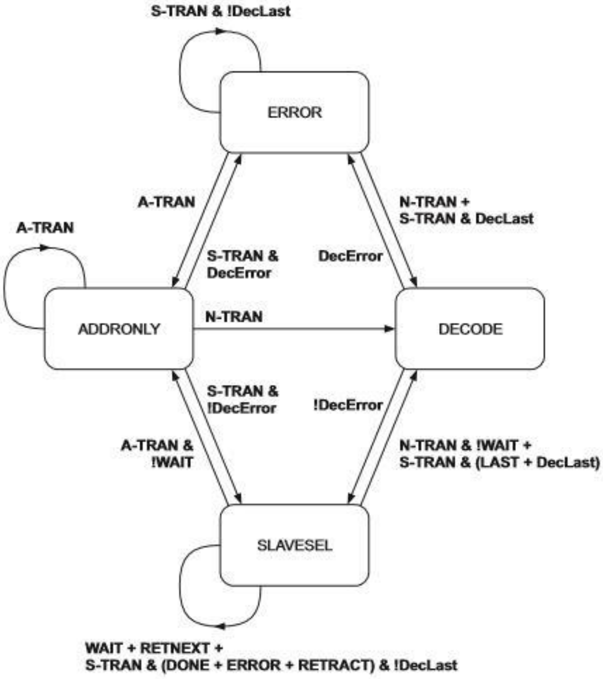
S-TRAN & !DecLast
ERROR
A-TRAN
N-TRAN + S-TRAN & DecLast
A-TRAN
S-TRAN & DecError
DecError
ADDRONLY
N-TRAN
DECODE
S-TRAN & !DecError
!DecError
A-TRAN & !WAIT
N-TRAN & !WAIT + S-TRAN & (LAST + DecLast)
SLAVESEL
WAIT + RETNEXT + S-TRAN & (DONE + ERROR + RETRACT) & !DecLast

Fig. 2. Decoder FSM with Decode cycle. [1]

D. Slave

A Slave owning the Bus has to respond with legit responses on BWAIT, BERROR, BLAST lines to the bus Masters. After sending an affirmative response for the received valid request, the Slave has to accept data on BDATA bus. Every Slave has a defined memory, and thus a Slave typically uses BLAST line to prevent a burst from continuing over a page boundary or other burst length limit. The Slave drives the error response on BERROR line in cases of Sequential Transfer when:

- the Master intends to write over a page boundary or,
- the address of current Sequential Transfer cycle is not related to that of previous cycle.

V. SIMULATION RESULTS

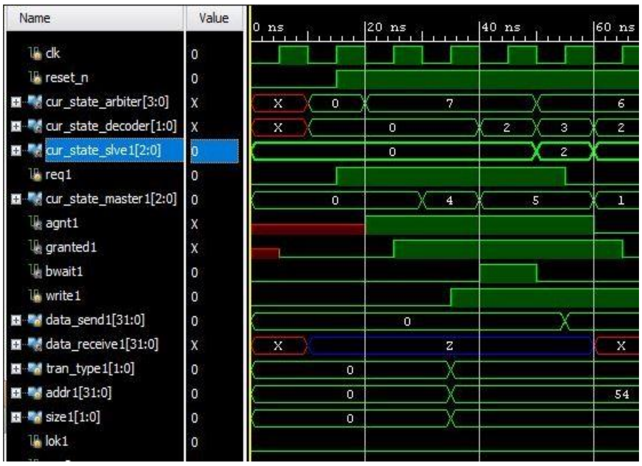

Fig. 3. Simulation results of ASB Non-Sequential Write Transfer Cycle

The design was simulated using Vivado Simulator by Xilinx. Figure 3 is the snapshot of simulation waveforms for the design describing Non-sequential type of write transfer. A Master has to keep the request signal asserted till the purpose of the bus is served. The Arbiter asserts AGNT signal if the Bus is free and keeps it asserted till the end of transaction. The current state of Arbiter's Finite State Machine denotes the next highest priority Master. The 'granted' signal in the figure, generated by the respective Master as a function of AGNT and BWAIT signals depicts that the respective Master is the current owner of the Bus.

The implemented design in the paper depicts an ASB Transfer protocol between 3 masters and 3 slaves. Figure 4 and 5 shows the RTL schematic for the Master-Arbiter interconnections and Slave-Decoder interconnections respectively.

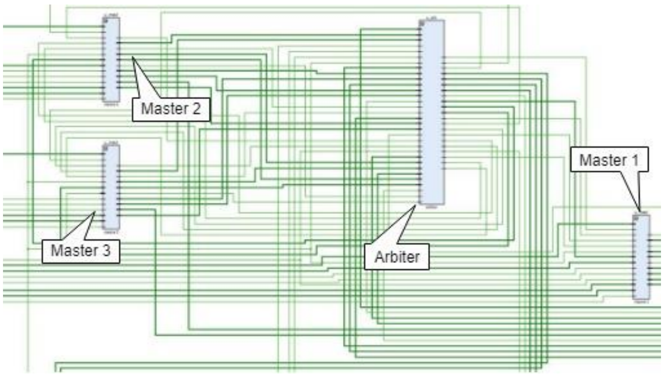

Fig. 4. Masters and Arbiter RTL schematic

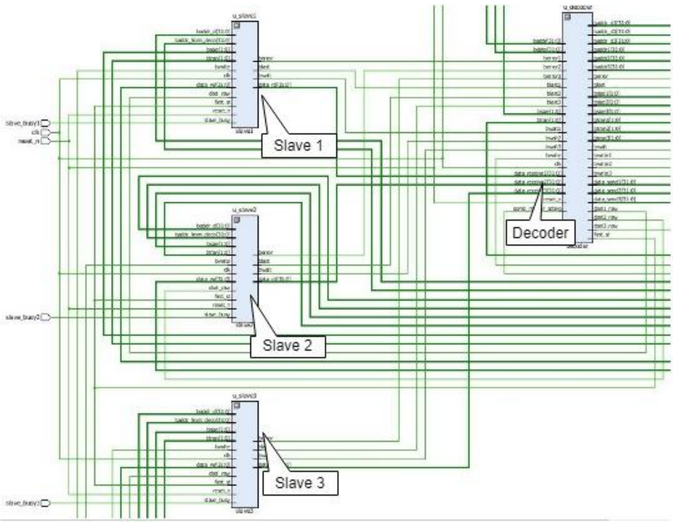

Fig. 5. Slave and Decoder RTL schematic

VI. CONCLUSION

The paper presents simulation of a synthesizable RTL design for AMBA Advanced System Bus protocol and that the design could be implemented solely based on the AMBA specification document published by ARM [1]. The design supports multi-master, multi-slave data transactions, however, since the higher order bits of address bus serve as decode lines for Slave, number of slaves limit the memory depth for each Slave. An assertion on AGNT pin from the Arbiter to each of the Masters in conjunction with BWAIT pin being low indicates that the Master has been granted the Bus, and only till the Bus is granted, a Master can drive the Bus to prevent contention or loss of data.

REFERENCES

[1] AMBA Specification (Rev 2.0), Copyright ARM Limited 1999.

[2] Verilog HDL by Samir Palnitkar, Prentice Hall 1996.

[3] Bn, Manu & Parameshwarappa, Prabhavathi. (2013). Design and implementation of AMBA ASB APB bridge. 234-238. 10.1109/iFuzzy.2013.6825442.

[4] Anshu Gaur, Piyush Sharma, Shiv Pratap Pandey, "HDL and timing analysis of AMBA AHB on FPGA platform", Control Automation & Power Engineering (RDCAPE) 2017 Recent Developments in, pp. 2227, 2017.

Ankit Berde was born in Mumbai, India, in 1995. He received the Bachelor of Engineering degree in Electronics and Telecommunication discipline from the University of Mumbai, India, in 2016. Digital VLSI Systems has been his major field of research. He is carrying an independent research work in implementation and analysis of different Bus Architecture.